Rでグラフ作成！

-基礎の基礎の入門編-

担当 : 河崎祐樹
森林保護 D2

なぜRでグラフを書くの？

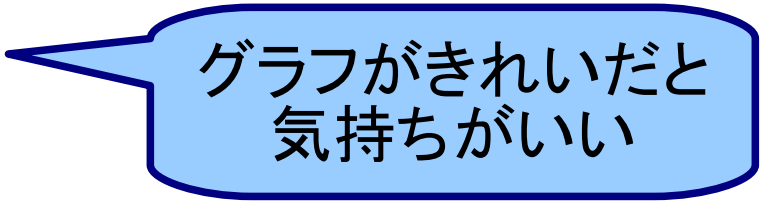
1. グラフがきれい

2. 書き直しが簡単

3. 同じようなグラフを
   簡単に書ける

なぜRでグラフを書くの？

1. グラフがきれい



2. 書き直しが簡単

3. 同じようなグラフを
   簡単に書ける

なぜRでグラフを書くの？

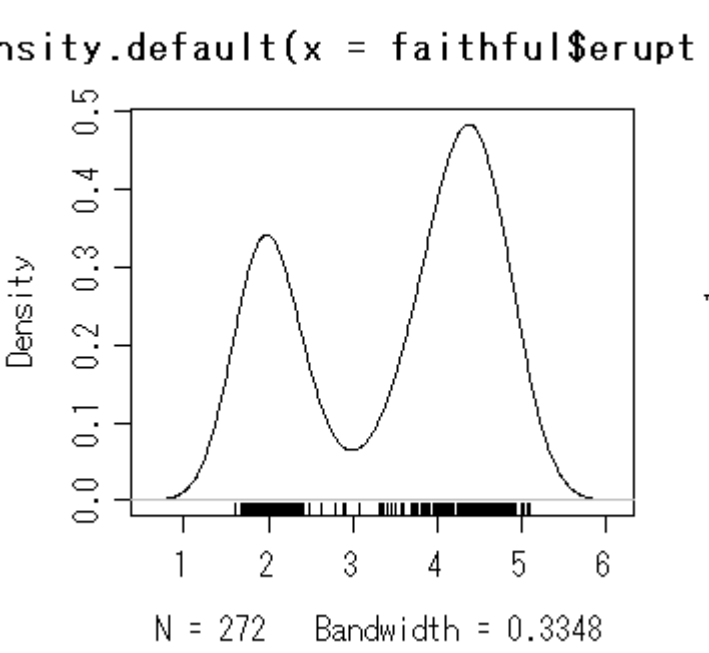
1. グラフがきれい

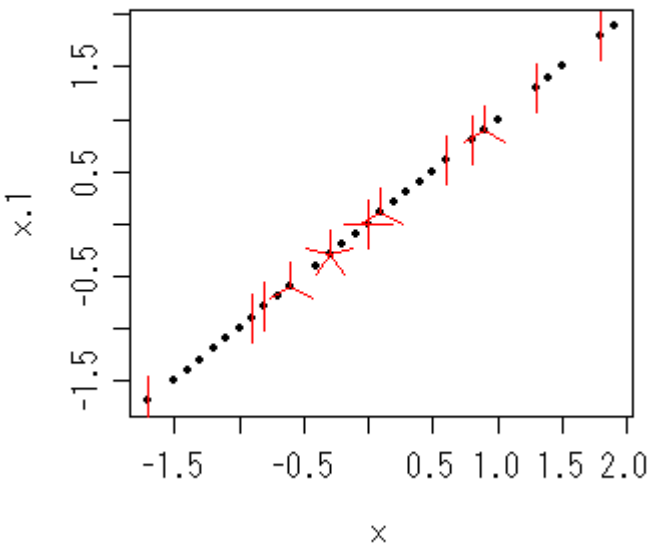
グラフがきれいだと
気持ちがいい

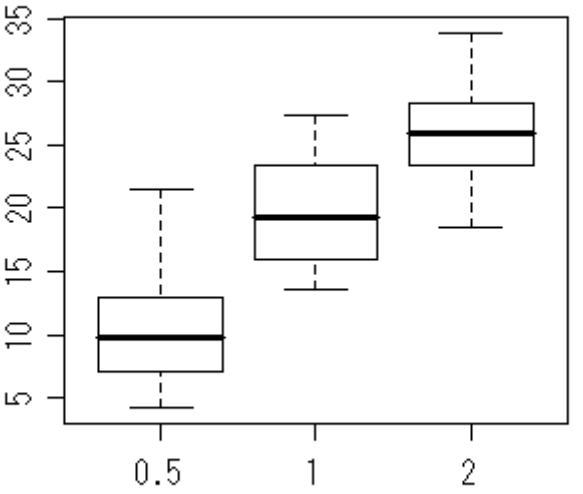
2. 書き直しが簡単

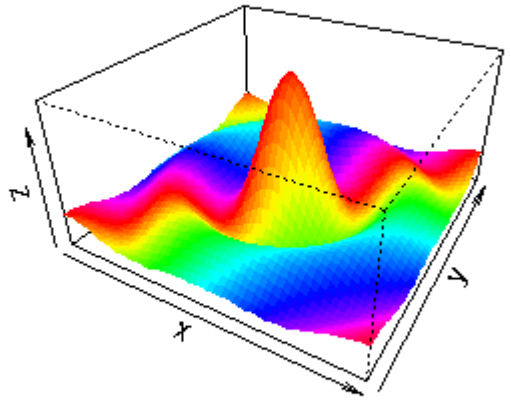
セミナー直前，卒論･修論
直前の書き直し命令
でもあわてない

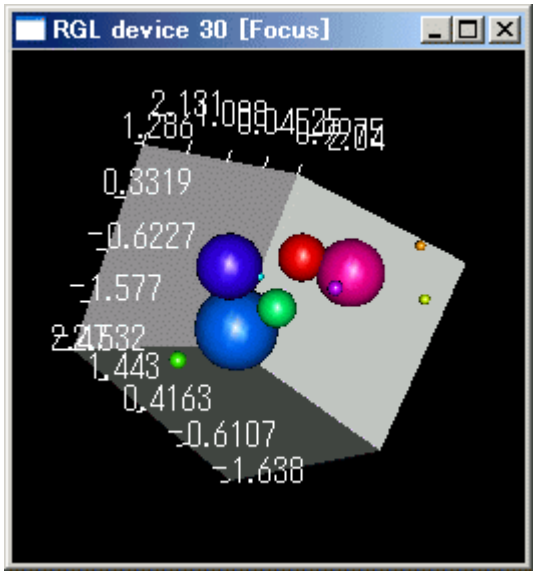
3. 同じようなグラフを
簡単に書ける

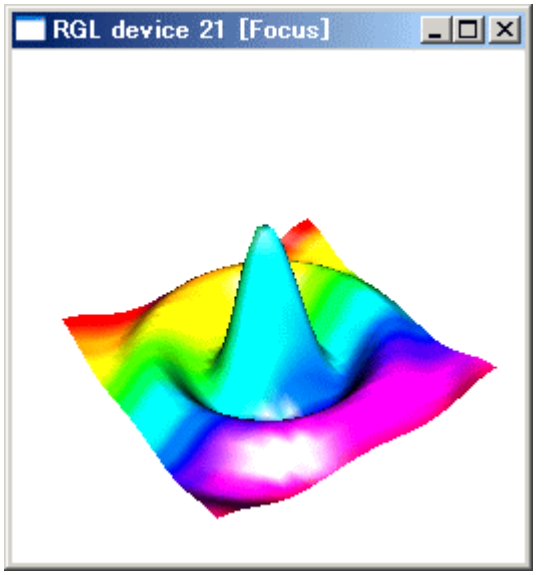

nsity.default(x = faithful$erupt
Density
0.0 0.1 0.2 0.3 0.4 0.5
1 2 3 4 5 6
N = 272 Bandwidth = 0.3348


x.1
-1.5 -0.5 0.5 1.5
-1.5 -0.5 0.5 1.0 1.5 2.0
x


5 10 15 20 25 30 35
0.5 1 2


z
x
y


RGL device 30 [Focus]
0.8319
-0.6227
-1.577
1.443
0.4163
-0.6107
-1.638


RGL device 21 [Focus]

なぜRでグラフを書くの？

1. グラフがきれい

グラフがきれいだと
気持ちがいい

2. 書き直しが簡単

セミナー直前，卒論･修論
直前の書き直し命令
でもあわてない

3. 同じようなグラフを
簡単に書ける

時間の節約！！

Rでグラフを書く前の準備

実験･観察･調査をする

↓

データを取る

↓

データをExcelに入力する

↓

･･･でも，RはExcel形式の
データを読み込めない

Rでグラフを書く前の準備

実験･観察･調査をする

↓

データを取る

↓

データをExcelに入力する

↓

・・・でも，RはExcel形式の
データを読み込めない

Rでグラフを書く前の準備

実験･観察･調査をする

↓

データを取る

↓

データをExcelに入力する

↓

･･･でも，RはExcel形式の
データを読み込めない

Rで読み込める形式に変えてあげる

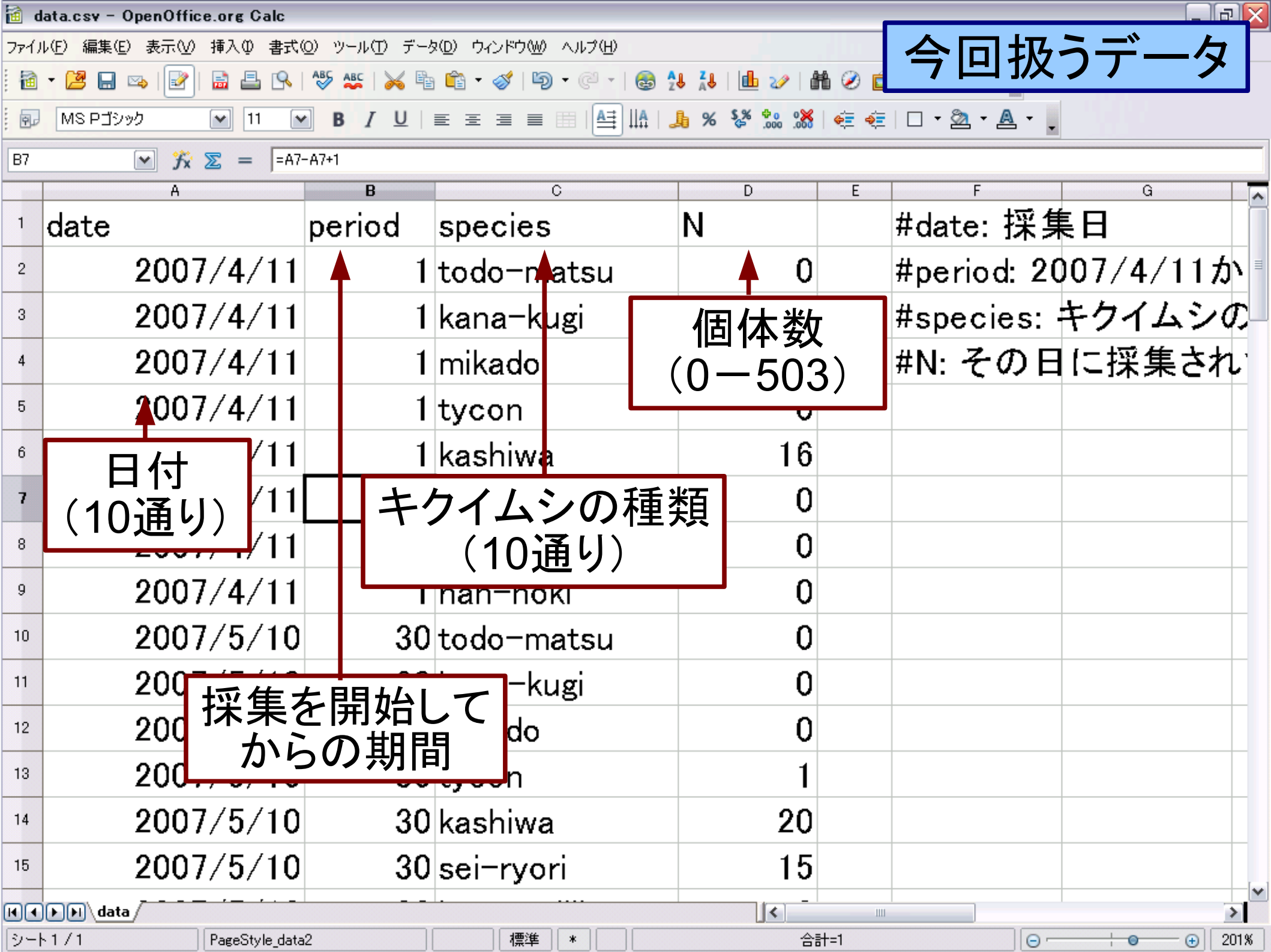
Rでグラフを書く前の準備

1.データファイルをExcelで開く
2.ファイル → 名前をつけて保存
　→ ファイルの種類
　　タブ形式（.txt）／csv形式（.csv）
3.保存する場所を選択
　マイコンピュータ → My Document → R

今回の目標
Rでグラフを描いてみたくなる

1. データを自由に扱えるようになる
   - 1-1 エクセルに保存したデータを読み込ませる
   - 1-2 読み込ませたデータから、必要な値を取り出せる
   - 1-3 条件をつけて、必要な値を取り出せる

2. 関数plot()が使えるようになる
   - 2-1 x軸とy軸に代入する値を指定できる
   - 2-2 グラフの色やプロットの形を指定できる
   - 2-3 グラフのタイプを変更できる
   - 2-4 グラフのタイトル、x軸やy軸に名前を入れられる
   - 2-5 一枚のシートに複数のグラフを重ねられる
   - 2-6 グラフにタイトル・凡例をつける
   - 2-7 グラフを保存できる



	A	B	C	D	E	F	G
1	date	period	species	N		#date: 採集日	
2	2007/4/11	1	todo-matsu	0		#period: 2007/4/11か	
3	2007/4/11	1	kana-kugi			#species: キクイムシの	
4	2007/4/11	1	mikado			#N: その日に採集され	
5	2007/4/11	1	tycon				
6	/11	1	kashiwa	16			
7	/11	1		0			
8	/11	1		0			
9	2007/4/11	1	han-noki	0			
10	2007/5/10	30	todo-matsu	0			
11	200	30	-kugi	0			
12	200	30	do	0			
13	200	30	tycon	1			
14	2007/5/10	30	kashiwa	20			
15	2007/5/10	30	sei-ryori	15			

１．データを自由に扱えるようになる

1-1 エクセルに保存したデータを読み込ませる
1-2 読み込ませたデータから、必要な値を取り出せる
1-3 条件をつけて、必要な値を取り出せる

1. データを自由に扱えるようになる

1-1 エクセルに保存したデータを読み込ませる

```
#ファイル
# → 名前をつけて保存
#  → ファイルの種類からタブ区切り.txtか
#                                  カンマ区切り.csvを選択
#   → 適当な名前・保存場所に保存

#Rにデータを読み込ませるときはread.table()/read.csv()を使う
read.table("ファイル名.txt", header = T)
read.csv  ("ファイル名.csv", header = T)
```

1. データを自由に扱えるようになる

1-1 エクセルに保存したデータを読み込ませる

```
#ファイル
# → 名前をつけて保存
#  → ファイルの種類からタブ区切り.txtか
#                                  カンマ区切り.csvを選択
#   → 適当な名前･保存場所に保存

#Rにデータを読み込ませるときはread.table()/read.csv()を使う
read.table("ファイル名.txt", header = T)
read.csv  ("ファイル名.csv", header = T)

#data.csvというデータファイルを読み込ませる
data <- read.csv("data.csv", header = T)
data  #読み込んだデータが表示される
```

1. データを自由に扱えるようになる

1-2 読み込ませたデータから、必要な値を取り出せる

```
#1-2
names(data)   #列の名前row namesを確認
```

1. データを自由に扱えるようになる

1-2 読み込ませたデータから、必要な値を取り出せる

```
#1-2
names(data)   #列の名前row namesを確認

#全体のデータのうち、個体数(N)だけ知りたい
data$N   #dataの中のN ($ = の)
```

1. データを自由に扱えるようになる

1-2 読み込ませたデータから、必要な値を取り出せる

```
#1-2
names(data)   #列の名前row namesを確認

#全体のデータのうち、個体数(N)だけ知りたい
data$N   #dataの中のN ($ = の)

#全体のデータのうち、どんな種類のキクイムシがいるかが知りたい
levels(data$species)
```

1. データを自由に扱えるようになる

1-2 読み込ませたデータから、必要な値を取り出せる

```
#1-2
names(data)   #列の名前row namesを確認

#全体のデータのうち、個体数(N)だけ知りたい
data$N   #dataの中のN ($ = の)

#全体のデータのうち、どんな種類のキクイムシがいるかが知りたい
levels(data$species)

#キクイムシの種数がほしい
nlevels(data$species)
```

1. データを自由に扱えるようになる

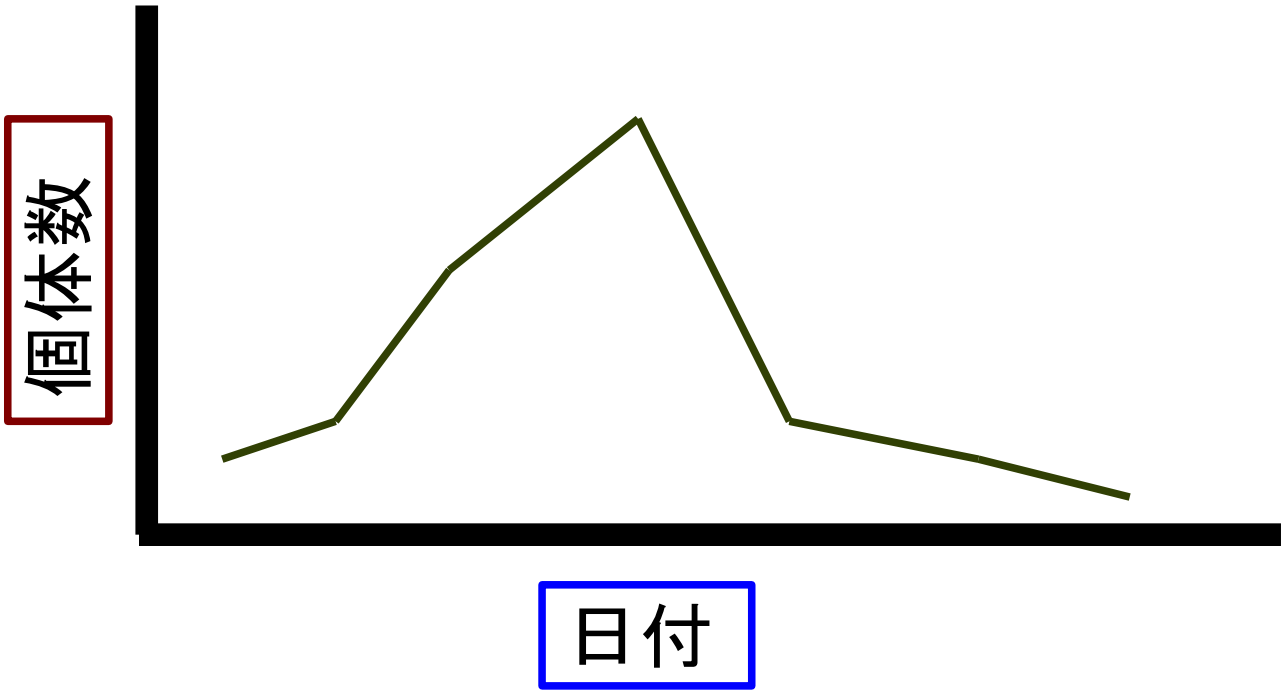
1-3 条件をつけて、必要な値を取り出せる

```
#全体のデータのうち、種speciesがhane-mijika
#という条件の個体数Nだけ知りたい
data$N[data$species == "hane-mijika"]
```

条件は[]で囲む。
== を2つつなげる
文字の場合は、””で囲む

2. 関数plot()が使えるようになる

2-1 x軸とy軸に代入する値を指定できる



こんなグラフを作るには・・・

X軸：日付のデータ

Y軸：個体数のデータ

を代入すればよい

２．関数plot()が使えるようになる

2-1 x軸とy軸に代入する値を指定できる
2-2 グラフの色やプロットの形を指定できる
2-3 グラフのタイプを変更できる
2-4 グラフのタイトル、x軸やy軸に名前を入れられる
2-5 一枚のシートに複数のグラフを重ねられる
2-6 グラフにタイトル・凡例をつける
2-7 グラフを保存できる

2. 関数plot()が使えるようになる

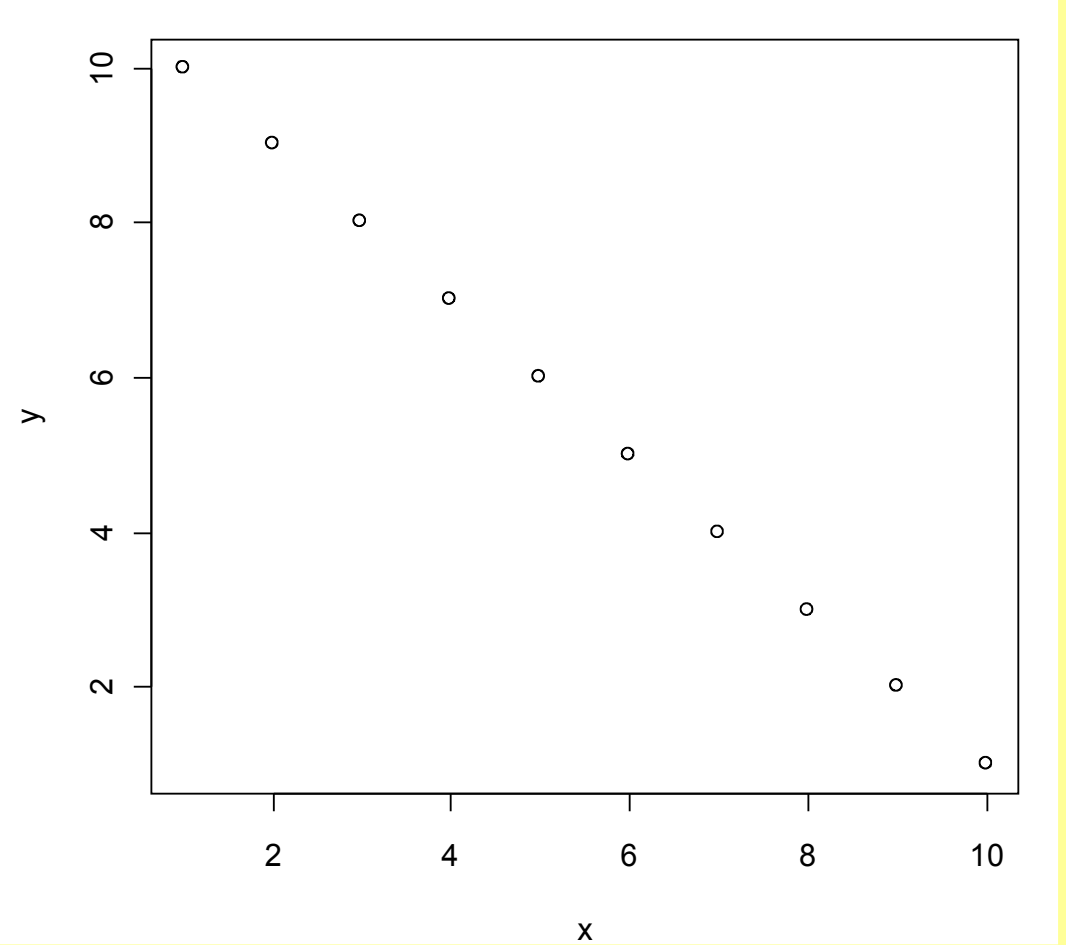
2-1 x軸とy軸に代入する値を指定できる

```
plot("X軸", "Y軸")

x <- c(1:10)#xに1, 2, 3, 4, 5, 6, 7, 8, 9, 10を代入
y <- c(10:1)#yに10, 9, 8, 7, 6, 5, 4, 3, 2, 1を代入

plot(x, y)
```

2. 関数plot()が使えるようになる

2-1 x軸とy軸に代入する値を指定できる

```
plot("X軸", "Y軸")

x <- c(1:10)#xに1, 2, 3,
y <- c(10:1)#yに10, 9, 8

plot(x, y)
```



2. 関数plot()が使えるようになる

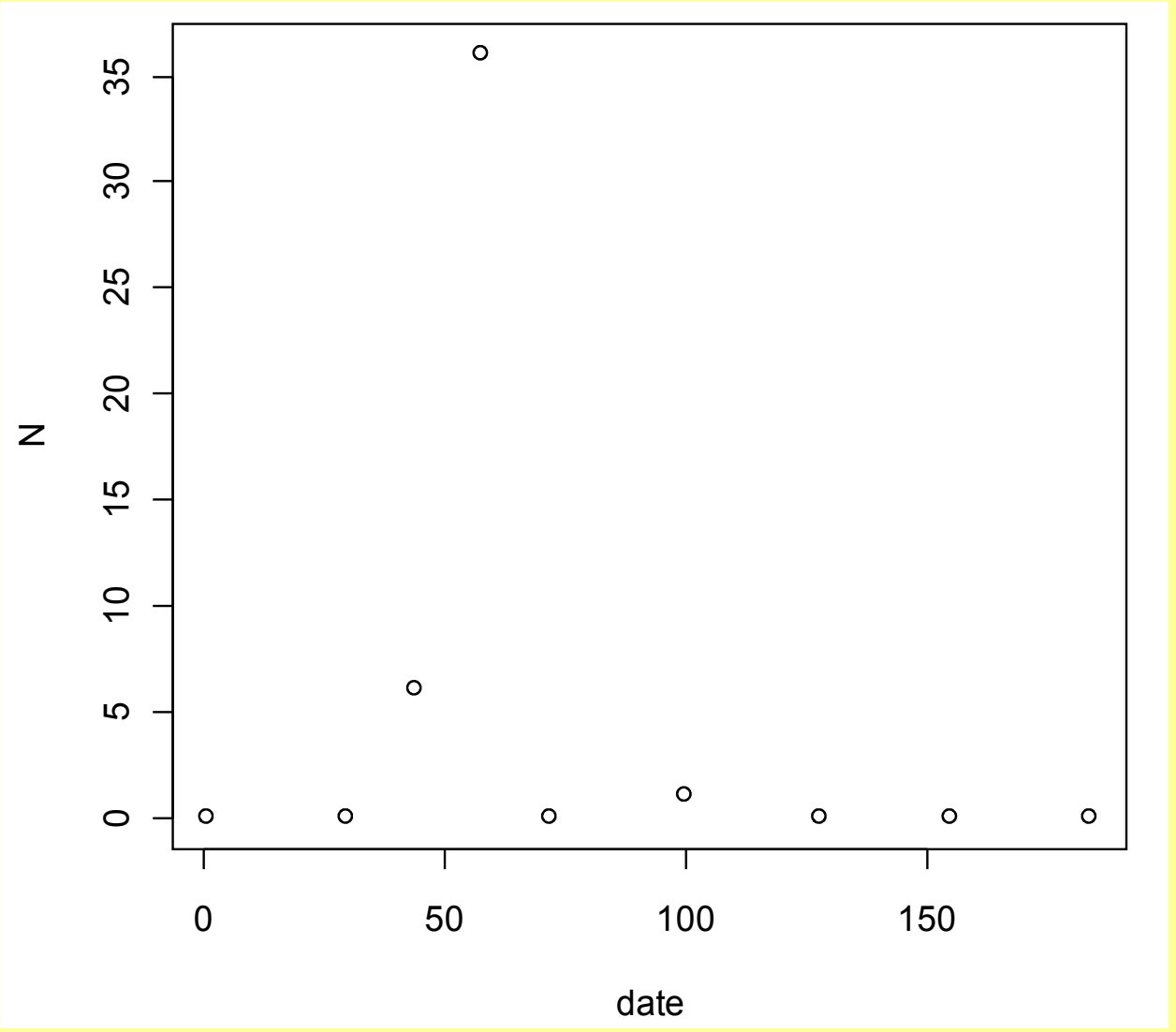
2-1 x軸とy軸に代入する値を指定できる

```
plot("X軸", "Y軸")

#hane-mijikaを採集した日付
date <- data$period[data$species == "hane-mijika" ]

#hane-mijikaの個体数
N <- data$N[data$species == "hane-mijika" ]

plot(date, N)
```

2. 関数plot()が使えるようになる

2-1 x軸とy軸に代入する値を指定できる

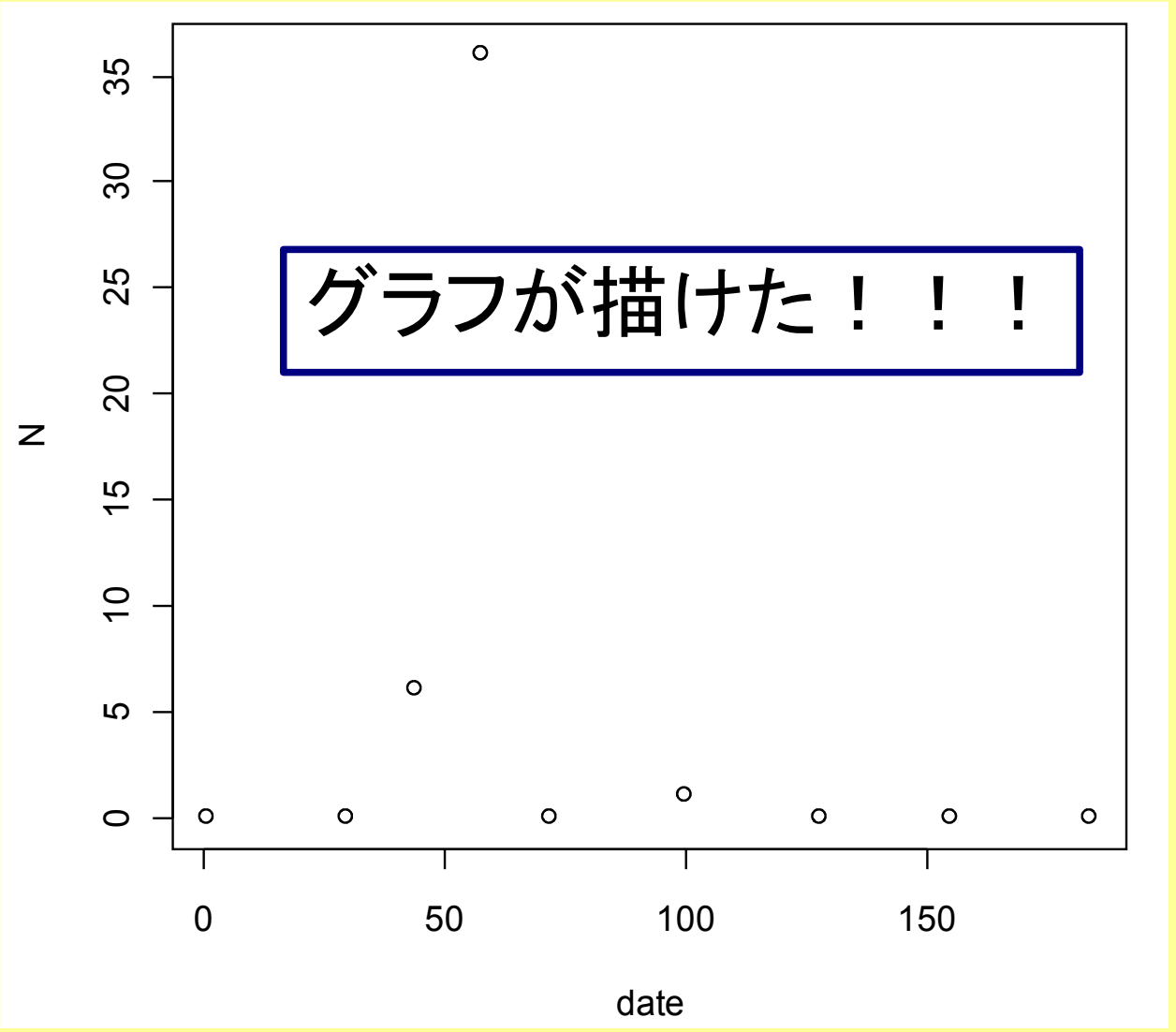
```
plot("X軸", "Y軸")

#hane-mijikaを採集した
date <- data$period

#hane-mijikaの個体数
N <- data$N[data$s

plot(date, N)
```



2. 関数plot()が使えるようになる

2-1 x軸とy軸に代入する値を指定できる

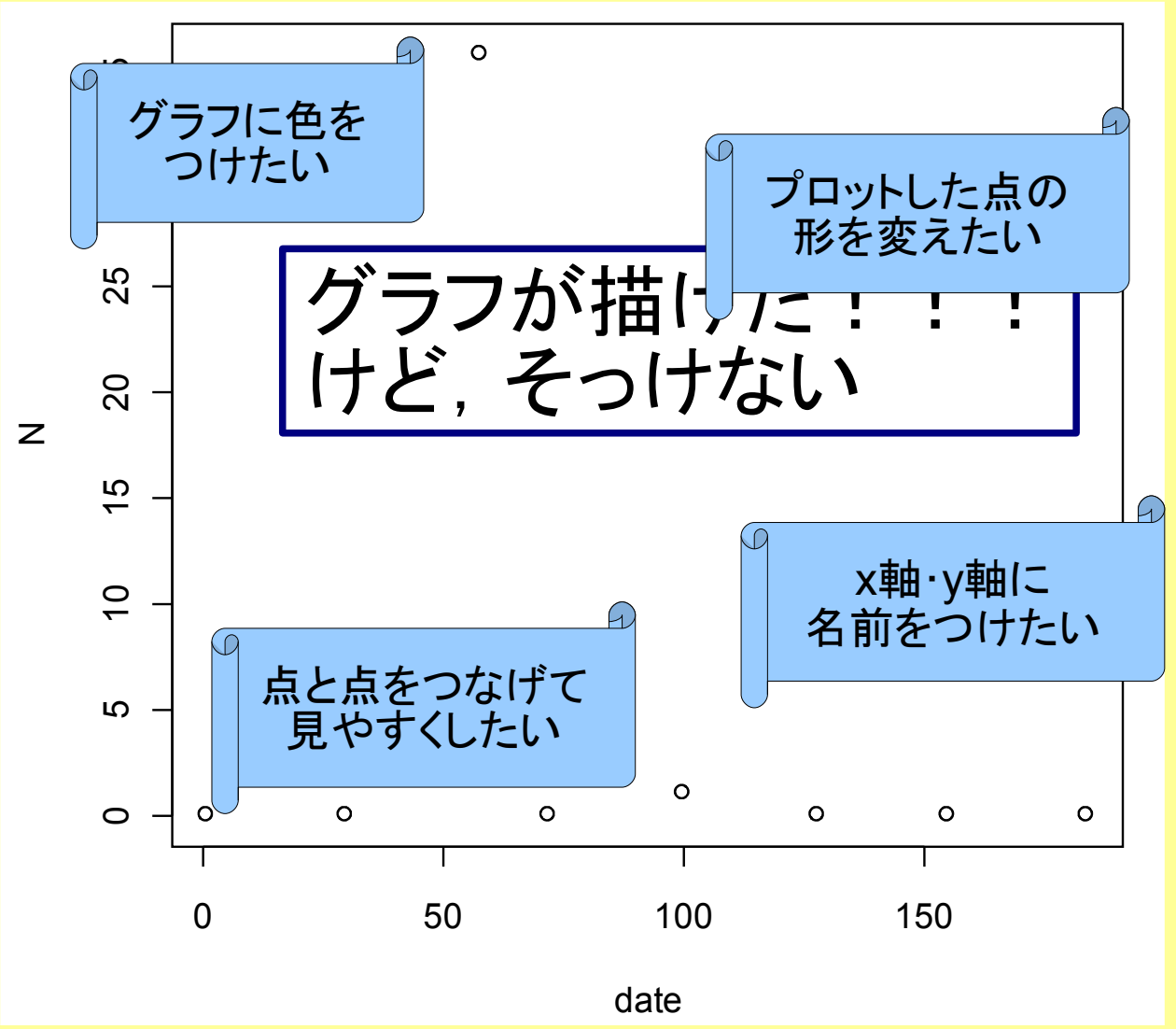
```
plot("X軸", "Y軸")

#hane-mijikaを採集した
date <- data$period

#hane-mijikaの個体数
N <- data$N[data$s

plot(date, N)
```



2. 関数plot()が使えるようになる

2-1 x軸とy軸に代入する値を指定できる

```
plot("X軸", "Y軸")

#hane-mijikaを採集した
date <- data$period

#hane-mijikaの個体数
N <- data$N[data$s

plot(date, N)
```



2. 関数plot()が使えるようになる

2-2 グラフの色やプロットの形を指定できる

```
#プロットの形
pch = 数字, pch = "文字"

plot(date, N, pch = 3)
plot(date, N, pch = 20)
plot(date, N, pch = "$")
plot(date, N, pch = "A")
```

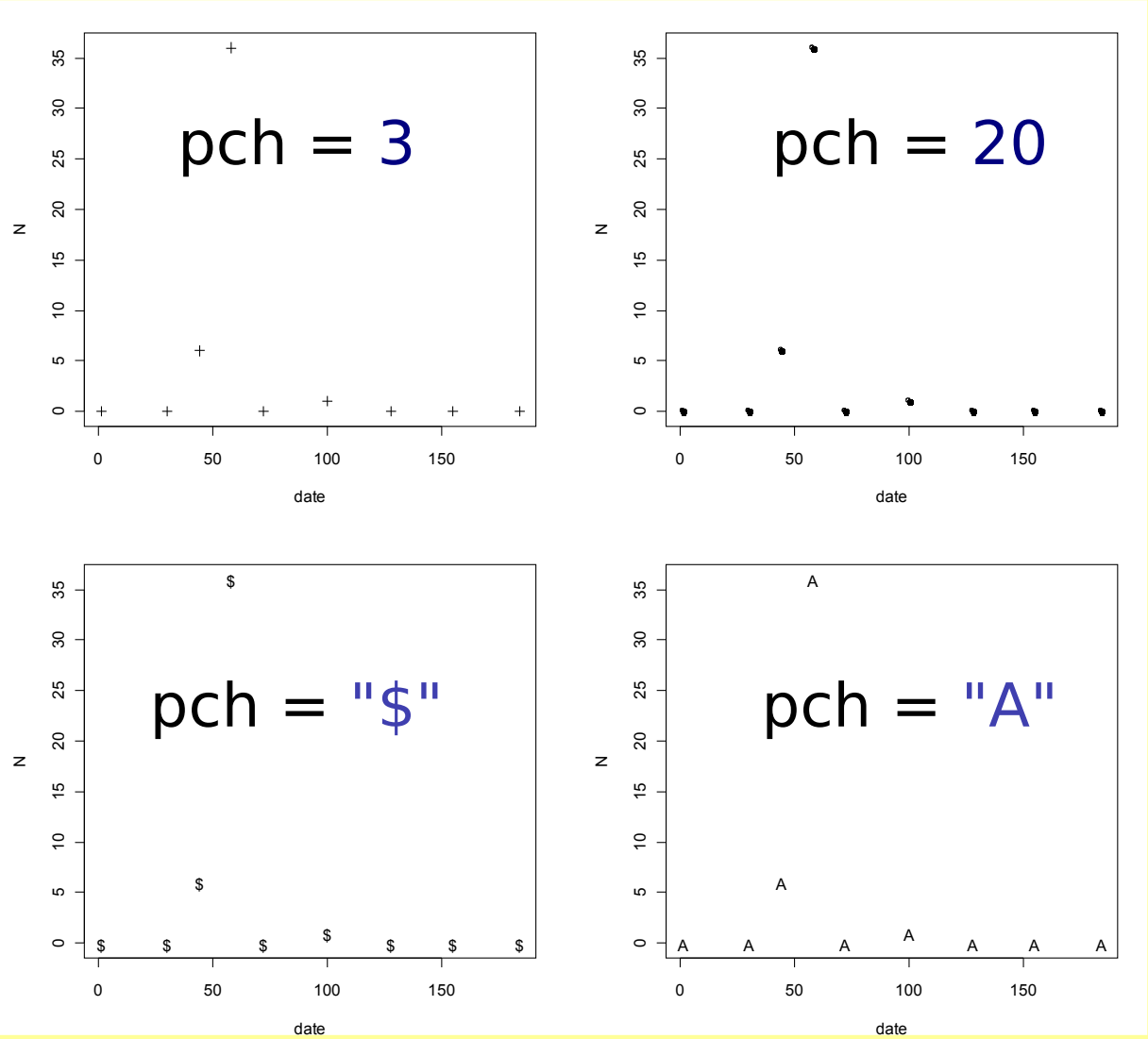
2. 関数plot()が使えるようになる

2-2 グラフの色やプロットの形を指定できる

#プロットの形

pch = 数字, pch = "文

plot(date, N, pch = 3
plot(date, N, pch = 2
plot(date, N, pch = "
plot(date, N, pch = "A



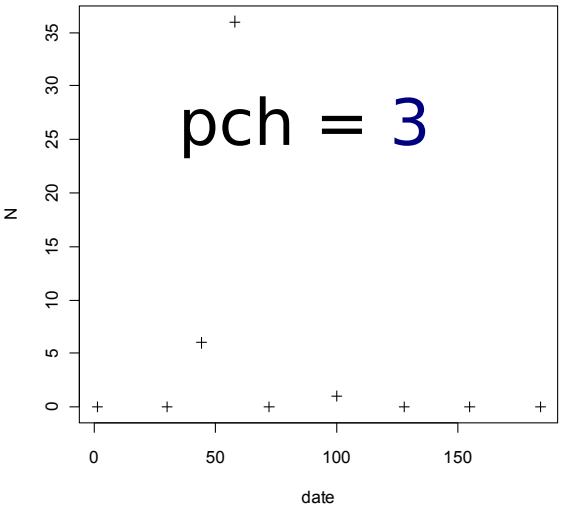
2. 関数plot()が使えるようになる

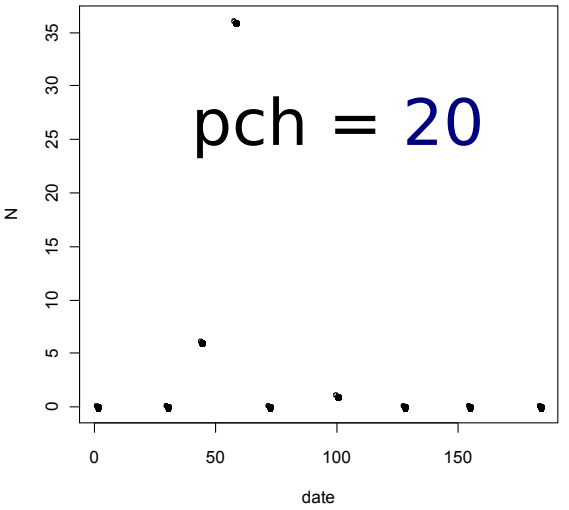
2-2 グラフの色やプロットの形を指定できる

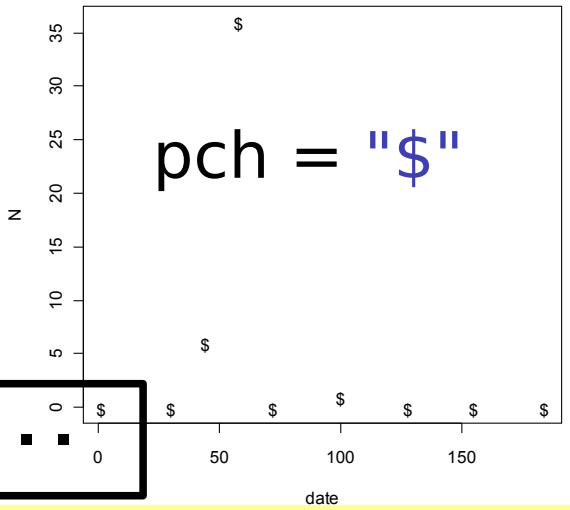
#プロットの形

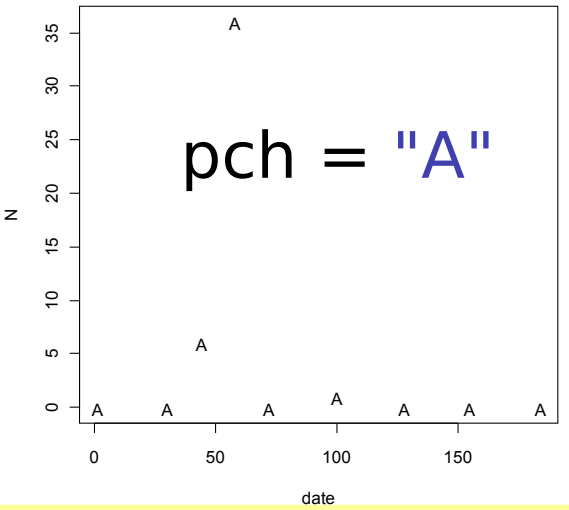
pch = 数字, pch = "文

plot(date, N, pch = 3
plot(date, N, pch = 2
plot(date, N, pch = "
plot(date, N, pch = "A









まだ、そっけない・・・

2. 関数plot()が使えるようになる

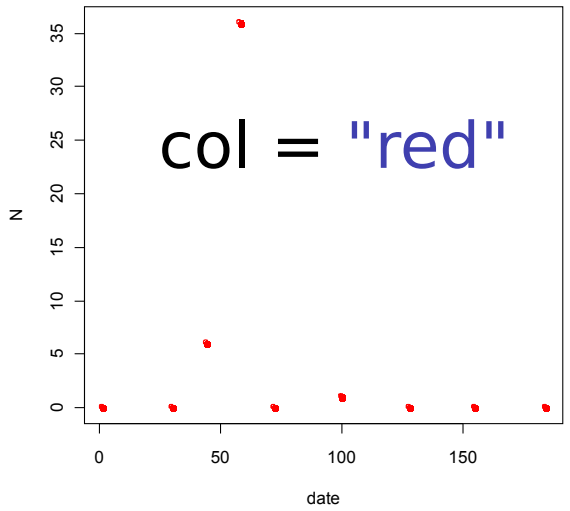
2-2 グラフの色やプロットの形を指定できる

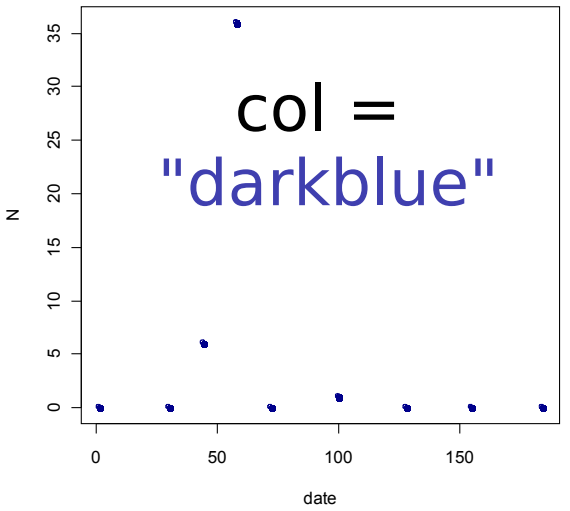
```
#色の変更
col = "color"
col = "数字"
##cf. colors

plot(date, N, pch = 20, col = "red")
plot(date, N, pch = 20, col = "darkblue")
plot(date, N, pch = 20, col = 3)
```

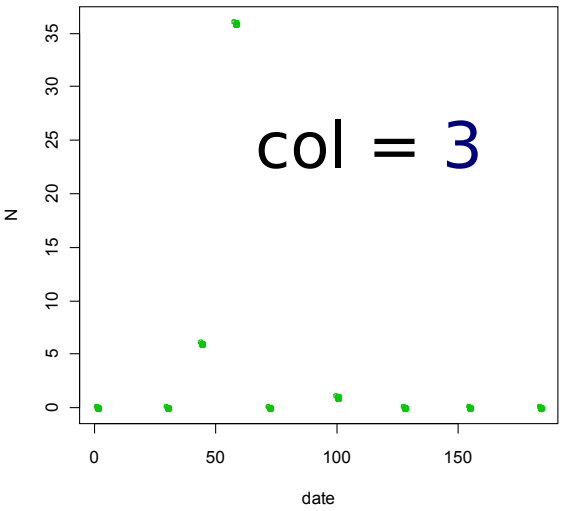
2. 関数plot()が使えるようになる

2-2 グラフの色やプロットの形を指定できる

```
#色の変更
col = "color"
col = "数字"
##cf. colors

plot(date, N, pch = 20, col = "red")
plot(date, N, pch = 20, col = "darkblue")
plot(date, N, pch = 20, col = 3)
```







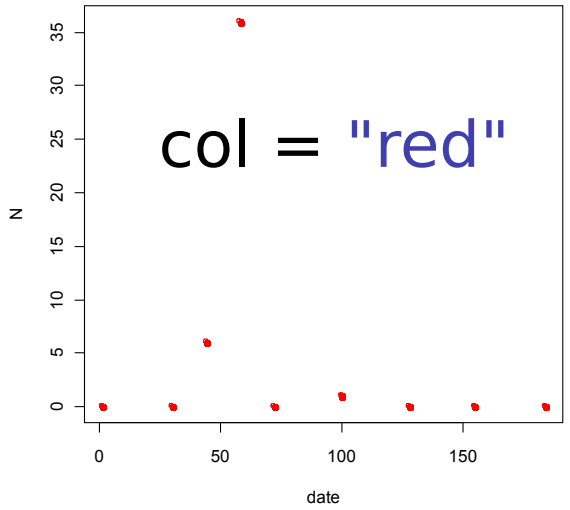
2. 関数plot()が使えるようになる

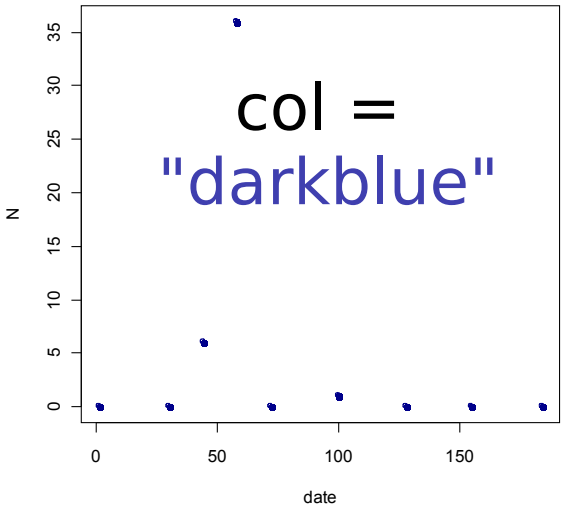
2-2 グラフの色やプロットの形を指定できる

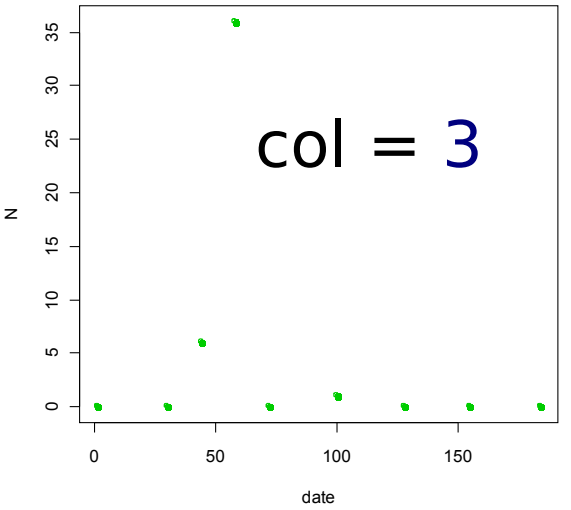
```
#色の変更
col = "color"
col = "数字"
##cf. colors
```

グラフの形がわかりにくい
点と点を線でつなげるには？

```
plot(date, N, pch = 20, col = "red")
plot(date, N, pch = 20, col = "darkblue")
plot(date, N, pch = 20, col = 3)
```







2. 関数plot()が使えるようになる

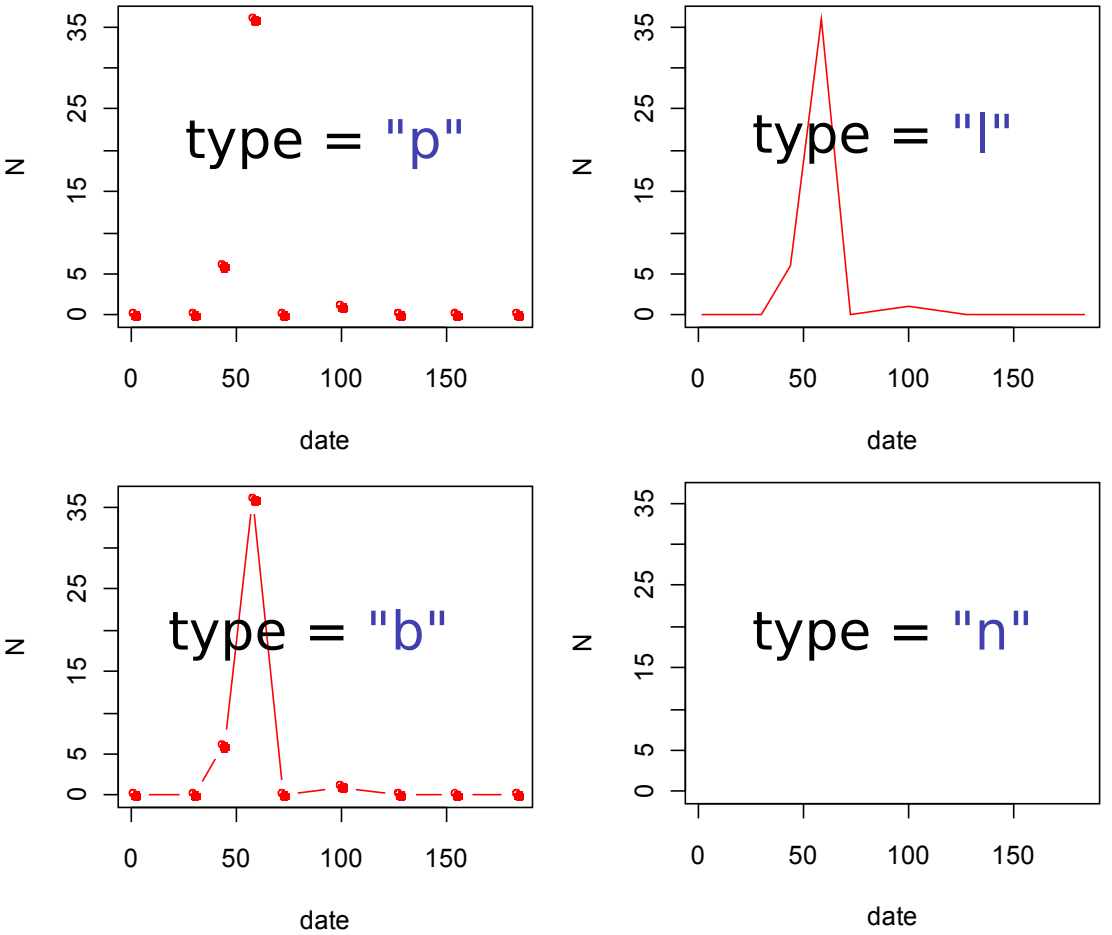
2-3 グラフのタイプを変更できる

```
#2-3
plot(x, y, type = "文字")
#p, l, b, c, o, h, s, S, n

plot(date, N, pch = 20,
      col = "red", type = "p")
plot(date, N, pch = 20,
      col = "red", type = "l")
plot(date, N, pch = 20,
     col = "red", type = "b")
plot(date, N, pch = 20,
      col = "red", type = "n")
```

2. 関数plot()が使えるようになる

2-3 グラフのタイプを変更できる

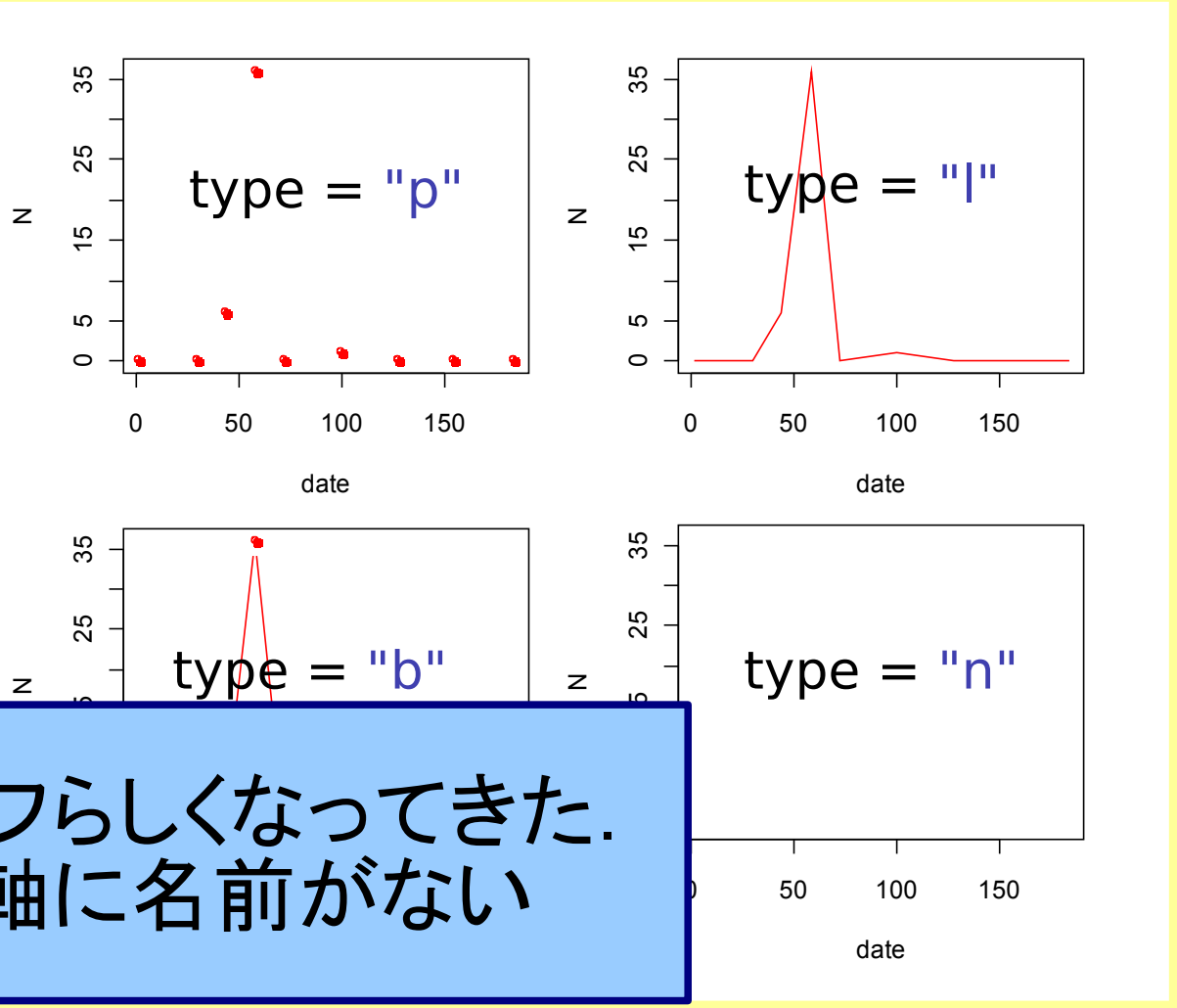
```
#2-3
plot(x, y, type = "文字"
#p, l, b, c, o, h, s, S, n

plot(date, N, pch = 20
      col = "red", type =
plot(date, N, pch = 20
      col = "red", type =
plot(date, N, pch = 20
     col = "red", type =
plot(date, N, pch = 20
      col = "red", type =
```



2. 関数plot()が使えるようになる

2-3 グラフのタイプを変更できる

```
#2-3
plot(x, y, type = "文字"
#p, l, b, c, o, h, s, S, n

plot(date, N, pch = 20
      col = "red", type =
plot(date, N, pch = 20
      col = "red", type =
plot(date, N, pch = 20
     col = "red", type =
plot(date, N, pch = 20
      co
```



だいぶ，グラフらしくなってきた．
・・・でも，軸に名前がない

2. 関数plot()が使えるようになる

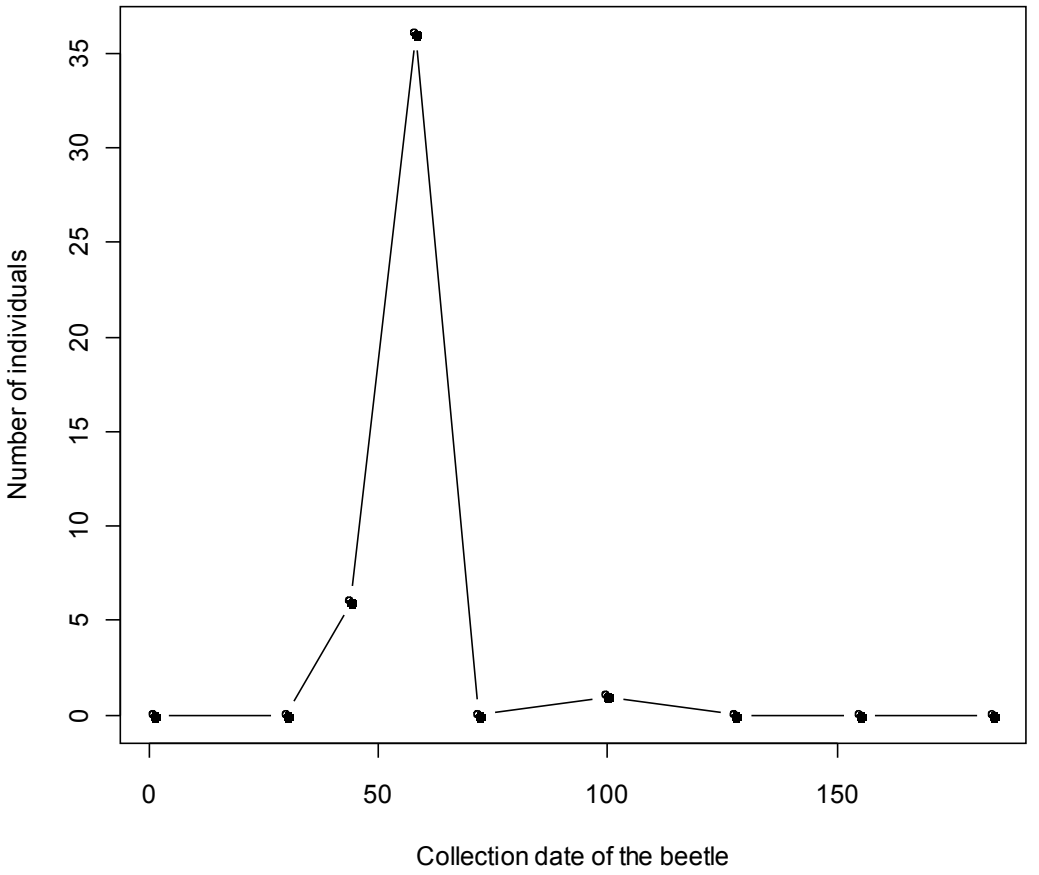
2-4 グラフのタイトル、x軸やy軸に名前を入れられる

```
#2-4
xlab = "X軸の名前"
ylab = "Y軸の名前"

plot(date, N,
     pch = 20,
     col = "black",
     type = "b",
     xlab = "Collection date of the beetle",
     ylab = "Number of individuals")
```

2. 関数plot()が使えるようになる

2-4 グラフのタイトル、x軸やy軸に名前を入れられる

```
#2-4
xlab = "X軸の名前"
ylab = "Y軸の名前"

plot(date, N,
     pch = 20,
     col = "black",
     type = "b",
     xlab = "Collection d
     ylab = "Number of i
```



2. 関数plot()が使えるようになる

2-5 一枚のシートに複数のグラフを重ねられる

```
#trap 1でとれたXylosandrus.germanusのLの日付
date <- data$pseudo.date[data$species == "hane-mijika"]
#trap 1でとれたXylosandrus.germanusのLの個体数
N <- data$N[data$species == "hane-mijika"]
```

2. 関数plot()が使えるようになる

2-5 一枚のシートに複数のグラフを重ねられる

```
#trap 1でとれたXylosandrus.germanusのLの日付
date <- data$pseudo.date[data$species == "hane-mijika"]
#trap 1でとれたXylosandrus.germanusのLの個体数
N <- data$N[data$species == "hane-mijika"]
```

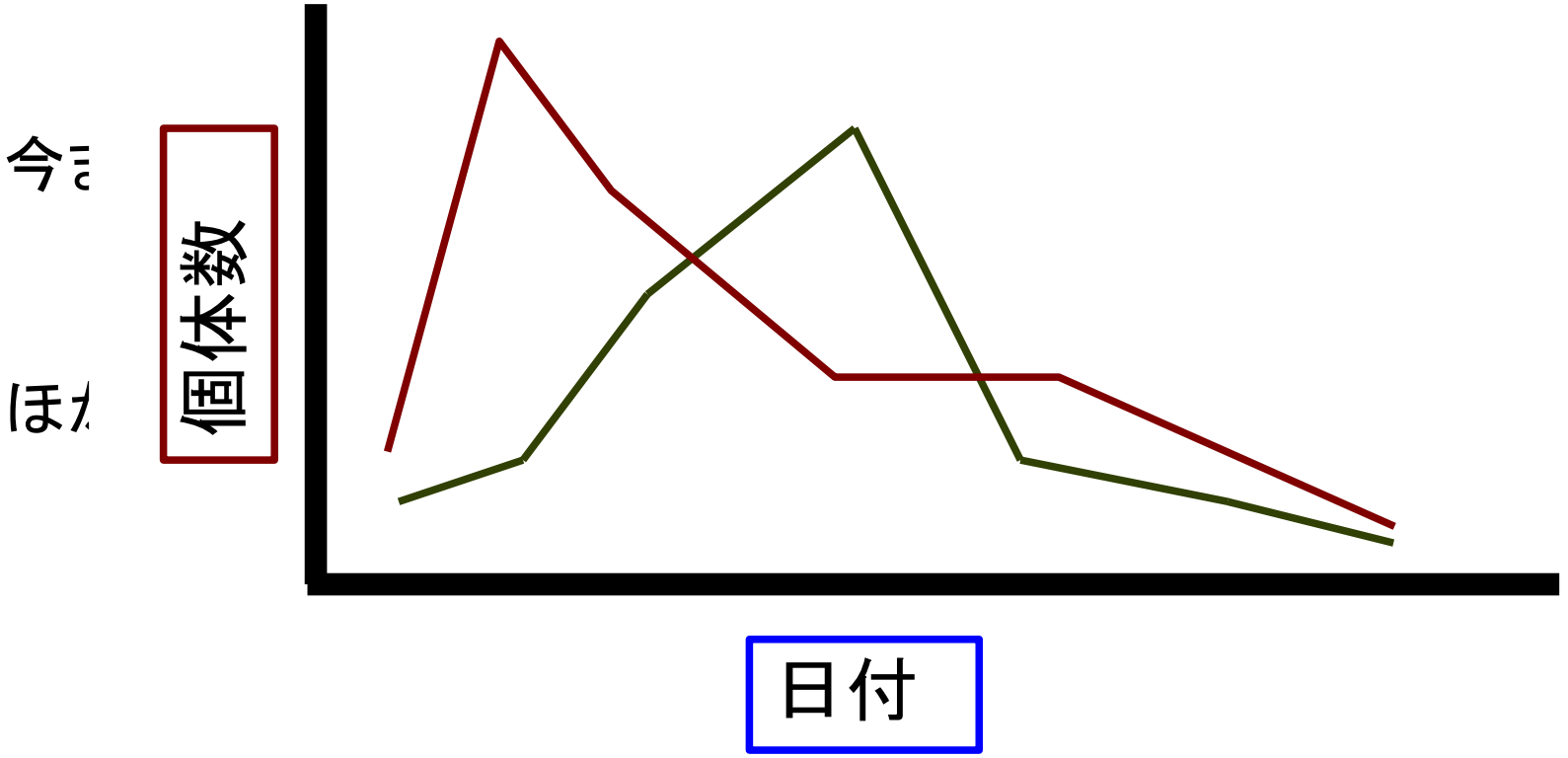
今までは、”hane-mijika”のグラフを書いてきた

ほかの種のグラフを重ねるには？

2. 関数plot()が使えるようになる

2-5 一枚のシートに複数のグラフを重ねられる

```
#trap 1でとれたXylosandrus.germanusのLの日付
date <- data$pseudo.date[data$species == "hane-mijika"]
#trap 1でとれたXylosandrus.germanusのLの個体数
N <- data$N[data$species == "hane-mijika"]
```



2. 関数plot()が使えるようになる

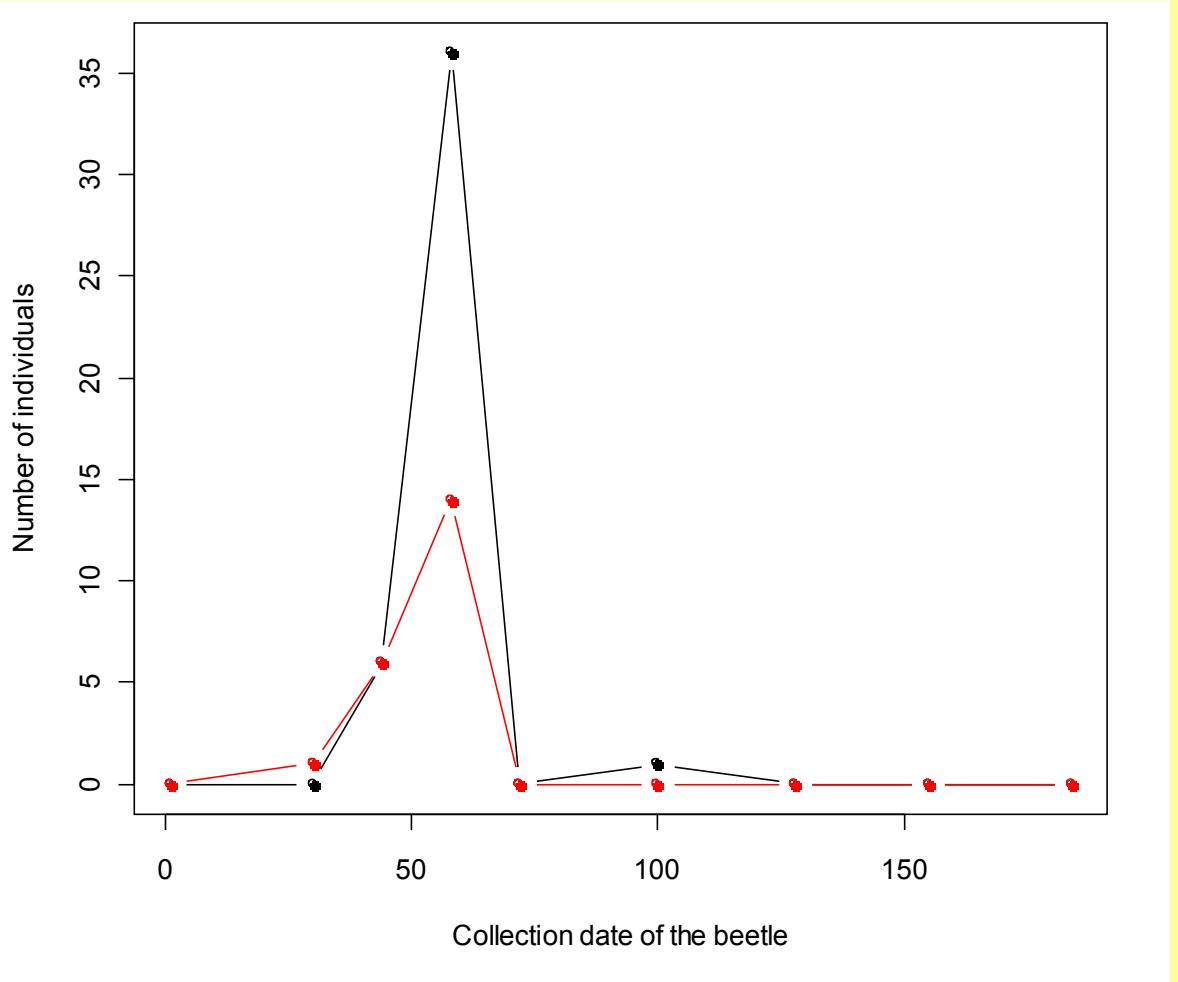
2-5 一枚のシートに複数のグラフを重ねられる

```
#2-5
#tyconのデータを読み込む
#"tycon"の日付
date_t <- data$period[data$species == "tycon"]

#tyconの個体数
N_t <- data$N[data$species == "tycon"]
```

2. 関数plot()が使えるようになる

2-5 一枚のシートに複数のグラフを重ねられる

```
#グラフを重ねるときはplotではなく、pointsを使う
points(date_t, N_t,
       pch = 20,
       col = "red",
       type = "b"
)
```



2. 関数plot()が使えるようになる

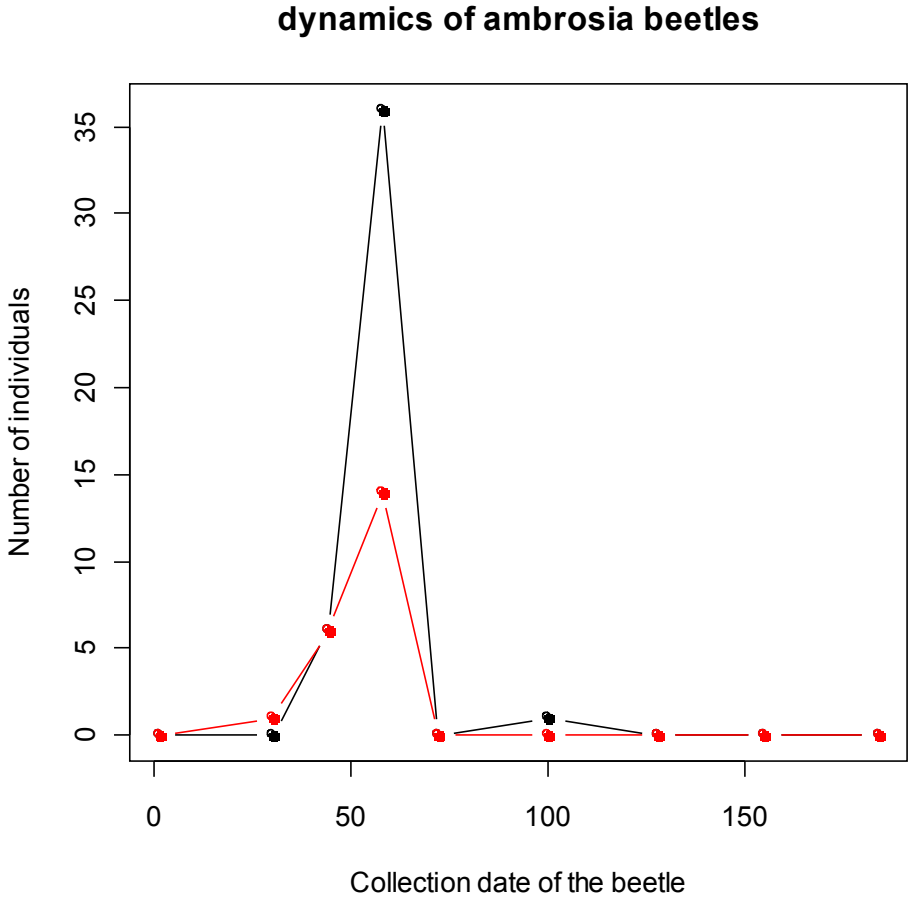
2-6 グラフにタイトル･凡例をつける

```
#plot()やpoints()でグラフを書いたあとに,
#title(main=″Main Title″, sub = ″Sub Title″)

title(main = "dynamics of ambrosia beetles")
```

2. 関数plot()が使えるようになる

2-6 グラフにタイトル・凡例をつける

```
#plot()やpoints()でグラ
#title(main="Main Title"

title(main = "dynam
```



2. 関数plot()が使えるようになる

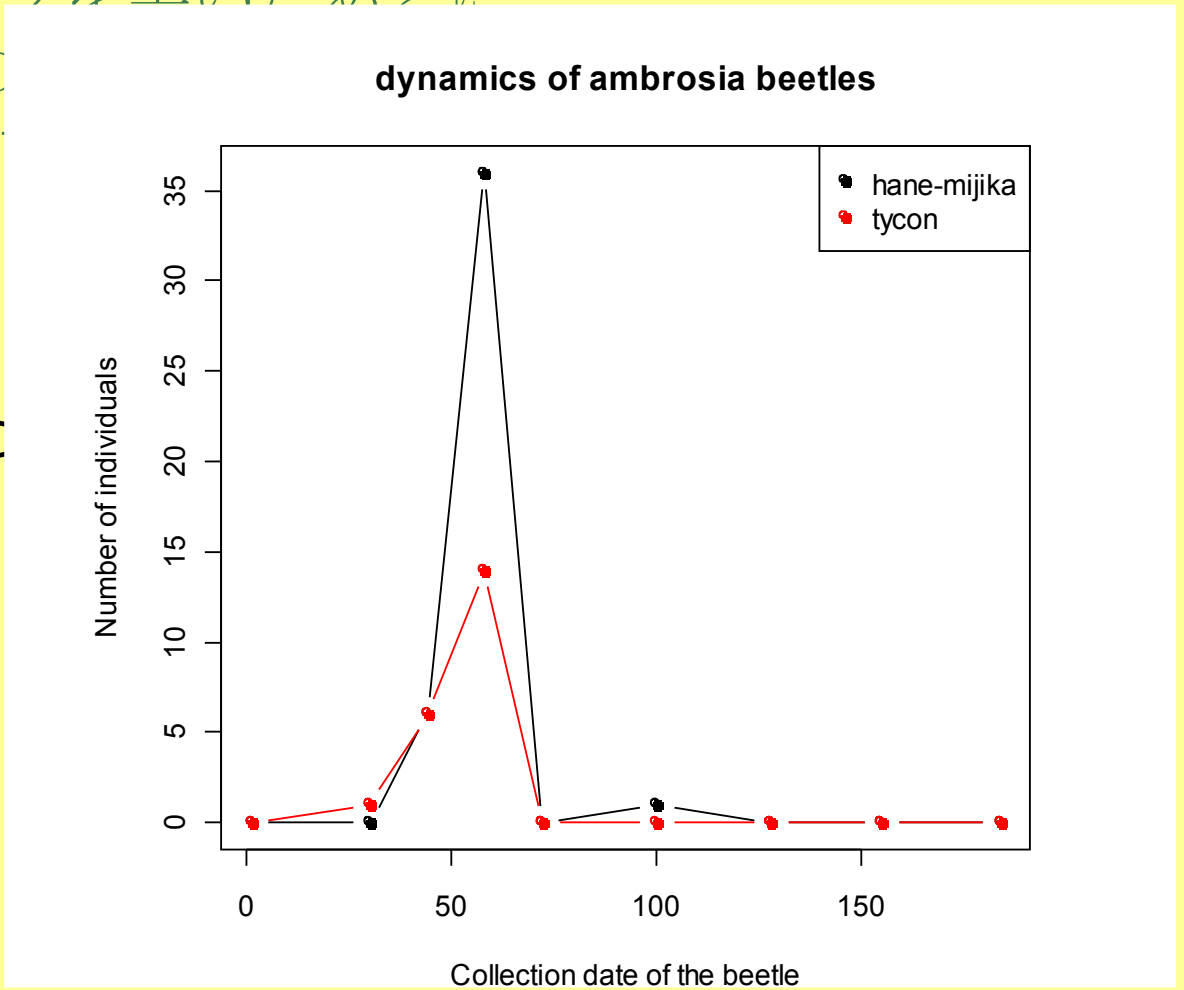
2-6 グラフにタイトル･凡例をつける

```
#plot()やpoints()でグラフを書いたあとに,
#legend(4, 4,  ← 凡例の位置
      paste("example",c(1:5)), ←凡例の名前
      col = c(1:5) ← 色
      )

legend("topright", c("hane-mijika", "tycon"),
    pch = 20
    col = c(1, 2)
)
```

2. 関数plot()が使えるようになる

2-6 グラフにタイトル･凡例をつける

```
#plot()やpoints()でグラフを書いたあとに
#legend(4, 4,  ← 凡例の
      paste("example",c(
      col = c(1:5) ← 色
      )

legend("topright", c
    pch = 20
    col = c(1, 2)
)
```



2. 関数plot()が使えるようになる

2-7 グラフを保存する

```
#2-7
#図の上で右クリック
#  → ○○でコピー
#  → イラストレーター、パワーポイントなどに貼り付け

#図の上で右クリック
#   → ○○で保存
#   → 保存場所・ファイル名を指定
```

2. 関数plot()が使えるようになる

2-7 グラフを保存できる

```
#PDFで保存することも可能
#pdf(file = ″ファイルネーム.pdf″)
#拡張子.pdfを忘れないように

pdf(file = "fig.pdf")
plot(date, N,
        pch = 20,
        col = "black",
        type = "b",
        xlab = "Collection date of the beetle",
        ylab = "Number of individuals")

points(date_t, N_t,
        pch = 20,
        col = "red",
        type = "b"
)
title(main = "dynamics of ambrosia beetles")
legend(”topright”, c(”hane-mijika”, ”tycon”), pch = 20, col = c(1, 2)
)
dev.off()   #PDFを閉じる
#忘れると、Rを閉じるまでPDFが開けない
```

まとめと補足

- グラフ用プログラムを書いて保存しておけば，
  書き直しも簡単

- plotを当てはめるデータを変更するだけで，
  似たようなグラフが簡単に描ける

- 今までに書いてきたプログラムが多ければ多いほど，
  グラフを描くのが楽になる

- 今回のグラフは初歩の初歩．
  卒業研究でよくやられる「年次変化」をグラフにする方法
  → 横軸を期間ではなく，日付にすることも可能（多少面倒）
  グラフを散布図・ヒストグラム・円グラフ・箱ひげ図
  などに変更も可能